REMARKS

The present invention is directed to a unique cache controller, and a method of controlling a cache to improve the efficiency of accessing and processing an execution of a plurality of tasks by dividing a cached memory into a plurality of regions and managing the specific regions so that when a hit/miss judgment decision in the execution of one of the tasks is performed, a search can be made by referring to all the cached memory addresses held in a cache directory. Reference can be made to the teachings in our specification on Page 26, Lines 18-21.

Thus, in the execution of a task by a microprocessor, even when data to be accessed is not stored in a specified region of a cache memory corresponding to that task, there could exist a possibility that the data may be stored in a region of the cache memory corresponding to another task. Since access time is reduced when dealing with a cache memory, our system enhances the performance of a hit/miss judgment unit in that policies provided to the hit/miss judgment unit can initiate a search of all the regions of the cache memory, and thereby enhance an aspect of a cache hit.

As can be appreciated, it is possible to achieve high speed processing by performing this hit/miss judgment in the execution of a task not by simply searching a region of the cache memory corresponding to that task, but by further searching all the regions of the cache memory, thereby increasing the probability of a cache hit.

Another benefit is achieved in performing the hit/miss judgment by searching all the regions of the cache memory when different tasks use the same memory address in the main memory. Our present invention would prevent a situation where different tasks use the same memory address from the main memory, and a plurality of entries corresponding to these tasks

could be generated in the cache memory, based on the same memory address of the main memory.

These features are now set forth in independent Claims 1, 14 and 15, and are distinguishable over the references of record.

The Office Action indicated that Claims 1-3 and 14-17 were rejected as being obvious over *Handy, The Cache Memory Book* when taken in view of the *Suh et al, Dynamic Cache Partitioning for Simultaneous Multithreading Systems* publication.

The Office Action further rejected Claims 4 and 7-11 over *Handy, Suh et al* and *Patterson et al, Computer Organization and Design* publication.

To provide a clarification to the record, Paragraph 5 on Page 5 of the Office Action, incorrectly cited the *Patterson et al* disclosure. As indicated in a short telephone conference on June 1, 2006 with Examiner Peers, the *Handy* reference was meant to be cited against Claims 8-11. Other than correcting this error, the prior art was not discussed, nor were the claims discussed with Examiner Peers.

The *Handy* textbook discloses a controller and includes a cache memory, a main memory and a microprocessor wherein the controller receives an address of a location in the main memory, for which data is to be accessed to execute one of the plurality of tasks in the microprocessor. The controller judges whether the data stored at the received address is stored in the cache memory, and if the data is not stored in the cache memory, acquires a data block including the data from the main memory, and stores the acquired data block into the cache memory.

As noted, initial storing of data would occur at a main memory operating speed until a loop would occur that could then be subsequently executed at the much faster cache speed.

Handy, however, does not disclose a technique of, in multitasking, managing the plurality of regions of the cache memory, in correspondence with the plurality of tasks and performing a specific hit/miss judgment in the execution of one of the tasks by searching all of the regions in the cache memory. As noted above, these features are set forth in each of the independent claims.

The *Suh et al* reference discloses a technique of dynamically dividing the cache for multithreading and assigning a divided area to each thread. While *Suh et al* is attempting to resolve a problem of "pollution," wherein a single thread can pollute the cache memory with its data and thereby cause higher miss rates for other threads, it does not address a technique of performing a hit/miss judgment in the execution of one of the threads by searching all the regions in the cache memory. Rather, *Suh et al* dynamically allocates parts of the cache to the most needy threads using online estimates of individual thread miss-rates.

The *Thaler et al* (U.S. Patent No. 5,983,329) was cited for certain features but did not address the lack of teaching of the primary *Handy* and *Suh et al* text relative to the currently pending claims.

Thus, *Thaler et al* may suggest a task identifier relative to an address of the data location in the main memory, but does not teach the structure of the cache controller currently defined in our current claims.

Likewise, the *Hum et al* publication 2002/0087824 is also directed to a manner of generating a task identifier by converting an address of a location in the main memory at which a task is stored as a program.

Finally, the *Chiou et al* paper on *Application-Specific Memory Management in Embedded Systems Using Software-Controlled Caches* suggests a scratchpad memory based on purportedly

a novel hardware mechanism called column caching. The column caching enables a dynamic cache partitioning in software by mapping the data regions into specific sets of columns. The column caching purportedly provides the same functionality and predictability as a dedicated scratchpad memory for time-critical parts of a real time application.

While *Chiou et al* may be broadly construed to divide a cache into specific columns to assist in executing each task within an application, it does not teach the features now defined relative to our judging step and judging unit of searching all the plurality of regions in the cache memory.

In summary, any hypothetical combination of some or all of the references relied upon to reject our claims, would fail to teach the features now set forth in our independent claims. Additionally, the further features set forth in our dependent claims would also suggest a *prima facie* case of obviousness has not been established.

It should be noted that the burden of establishing a *prima facie* case of obviousness lies with the Patent Office. *In re Fine*, 5 U.S.P.Q.2d 1596 (Fed. Cir. 1988) (stating: "The PTO has the burden under section 103 to establish a *prima facie* case of obviousness"). To establish a *prima facie* case of obviousness, (1) there must be some suggestion or motivation (either in the references themselves or in the knowledge generally available to one of ordinary skill in the art) to combine the reference teachings; (2) there must be a reasonable expectation of success; and (3) the prior art reference must teach or suggest all the claim limitations. See *MPEP* §§ 2142-43.

It is believed that the amendments to the claims now more than adequately establishes the patentability of our present invention and early notification of allowance is requested.

If the Examiner believes that a telephone interview will help further the prosecution of the case, the undersigned attorney can be contacted at the listed telephone number.

I hereby certify that this correspondence is being transmitted via facsimile to the USPTO at 571-273-8300 on June 19, 2006.

By: Sharon Farnus

Signature

Dated: June 19, 2006

Very truly yours,

SNELL & WILMER L.L.P.

Joseph W. Price
Registration No. 25,124
600 Anton Boulevard, Suite 1400
Costa Mesa, California 92626-7689
Telephone: (714) 427-7420
Facsimile: (714) 427-7799